LA-UR-14-21084
Approved for public release; distribution is unlimited.

Title: Thermal Analysis of Buried Flanged Tritium Waste Containers with Internal Heating

Author(s): Shimek, Moss
Lam, Kin

Intended for: FTWC Thermal Loading project record.
Report

Issued: 2014-02-20

Disclaimer:
Los Alamos National Laboratory, an affirmative action/equal opportunity employer,is operated by the Los Alamos National Security, LLC for the National NuclearSecurity Administration of the U.S. Department of Energy under contract DE-AC52-06NA25396. By approving this article, the publisher recognizes that the U.S. Government retains nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Departmentof Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

Thermal Analysis of Buried Flanged Tritium Waste Containers with Internal Heating

Moss Shimek, W-13

Kin Lam, W-13

Los Alamos National Laboratory

1 Introduction

This report documents the analysis requested for a Flanged Tritium Waste Container (FTWC). The objective of the analysis is to determine the effects of thermal loading due to radioactive decay of the maximum allowable amount of tritium per container. The LANL Waste Acceptance Criteria (WAC), P930-1, limits containers with high-activity tritium waste to a maximum of 100,000 Ci per package [1]. Tritium has an activity of 10,000 Ci/g, so 10 g represents the maximum amount in a container. The FTWC is constructed of stainless steel, overpacked inside an 85-gallon stainless steel drum, and buried at TA-54. Material specifications and dimensions for the FTWC and drum are given in references [2] and [3]. Figure 1 shows a schematic model of the FTWC system, in which the container, support skirt and overpack drum are modeled as right cylinders.

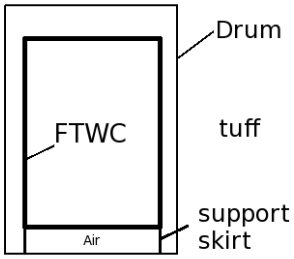

Figure 1: Schematic model of FTWC system consisting of the container, skirt, drum, and surrounding tuff.

The WAC also limits the thermal power output from buried containers. It sets a limit of 3.5 watts per cubic meter. Tritium generates 0.324 W/g from β-decay. Ten grams of tritium produces 3.24 W and the volume of the drum is 0.32 m^3, resulting in a rate of 10.12 watts per cubic meter. The question is what effects this thermal output has on the FTWC and the surrounding as-buried environs.

This analysis employs two complementary models to predict the temperature within the FTWC. A steady-state model is used to calculate the change in temperature from the drum's outer surface to the inside of the FTWC for a given power level. This heat conduction path is all stainless steel and has a time scale in minutes. A time-dependent finite-element model is used to predict the temperature rise in the surrounding volcanic tuff over the years as heat is conducted away from the drum. Our analysis takes into account reduction of the heat source magnitude due to the radioactive decay of tritium, which has a 12.32-year half life.

We have also estimated the pressure build-up in the FTWC as a result of gas generation from radiolysis in the waste in tritiated water form. The calculated pressure will be used to evaluate the structural integrity of the FTWC under as-buried conditions.

2 Steady-State Model

A steady-state heat transfer analysis is used to determine the temperature change from the outside of the drum to the inside of the FTWC. Heat generated in the FTWC is assumed to be uniformly distributed but conducted only through the radial wall. Similarly, conduction through the overpack drum is also assumed to be in the radial direction only. The FTWC is supported by a skirt (see figure 1) which we assume conservatively to be the only heat flow path out of the container to the overpack drum. Simple analytical solutions exist for one-dimensional steady-state heat conduction. The planar solution can be used for heat conduction through the skirt whereas the cylindrical 1D solution can be used for the FTWC and drum. The material properties and equations for the temperature change can be found in the textbook by Incropera and DeWitt [4]. The planar solution is

$$\Delta T = q\frac{\Delta X}{kA} \tag{1}$$

and the cylindrical solution is

$$\Delta T = q\frac{\ln(r_2/r_1)}{2\pi Hk} \tag{2}$$

where q is the power generated from radioactive decay, k is the thermal conductivity, ΔX is the distance the heat travels, A is the cross-sectional area of heat flow, r_2 and r_1 are the radial distances for cylindrical heat transfer, and H is the height of the FTWC or drum. Table 1 shows the calculated temperature change across each of the model components and the total value. This calculation assumes q to be at the maximum initial value of 3.24 W. As the tritium is depleted, q, and hence ΔT will decrease with the same exponential rate of decay.

All materials are stainless steel

	Material	r_1 (m)	r_2 (m)	H (m)	ΔX (m)	A (m^2)	k (W/m-K)	ΔT (°C)
Drum	304	0.330	0.332	0.997	NA	NA	14.9	0.00014
Skirt	316	NA	NA	NA	0.130	0.019	13.4	1.68
FTWC	316	0.308	0.318	0.707	NA	NA	13.4	0.0017
Total								**1.68**

Table 1: Steady-state temperature change results for initial power of 3.24 W.

Note that the temperature change across the drum is negligible (∼1E-4°C) because of the large surface area available for conduction and its thin wall. The FTWC has a thicker wall and a smaller surface area but the resulting ΔT is still quite small(∼1E-3°C). The dominant resistance to heat transfer is through the support skirt, because the heat has to travel a relatively long distance, ΔX, across a small cross-sectional area, A. Therefore, the total temperature change of 1.68°C across the FTWC system is essentially that between the top and bottom of the skirt.

3 Transient Model

3.1 Temperature Change

The steady-state model discussed above captures the temperature change between the FTWC and the drum. The drum is in contact with the surrounding tuff, which serves as the ultimate heat sink. To determine how much the FTWC heats up, we also need to find out how much the tuff next to the drum heats up.

An axisymmetric finite-element model was developed to predict the temperature of the tuff over time. It was created and simulated using the commercial finite-element analysis software package Abaqus [5]. The model assumes an infinitely long cylinder with the same diameter as the drum heating the surrounding tuff. This is a conservative model for a single FTWC because conduction in the axial direction is ignored. The model can also be considered to represent multiple FTWCs stacked on top of one another in a shaft. The initial thermal load into the tuff is a heat flux in the radial direction of 1.57 W/m^2, which decays with the 12.32-year half life of tritium. This heat flux value was determined by the total power of 3.24 W over the radial surface area of the drum.

	Thermal conductivity (W/m-K)	Density (kg/m^3)	Specific heat (J/kg-K)
Partially welded tuff	0.2	1800	1000

Table 2: Physical properties of tuff used in finite-element model.

One of the uncertainties in this analysis is the physical properties of tuff, especially the thermal conductivity. We used the physical properties of partially welded tuff as found in a 2006 paper by LANL volcanologist Ken

Wohletz [6], see table 2. Note that the thermal conductivity value of 0.2 W/m-K is the minimum given in the paper, and we choose that for the analysis to be conservative. As a comparison, a Sandia National Laboratories report by Lappin [7] discusses specifically the thermal conductivity of silicic tuff and the minimum value found in that reference is 0.6 W/m-K.

This model predicts the heat up (ΔT) in the tuff at 1-m increments away from the drum out to 50 m. All parameters used in the finite-element model can be found in the Abaqus input file listing in appendix A. The total ΔT in the FTWC above ambient is then given by adding the ΔT in the tuff adjacent to the drum calculated from this model and the ΔT across the drum/skirt/FTWC structure calculated from the steady-state model. Both models use a decreasing heat source governed by the half-life of tritium. Post-processing and combination of results from the transient and steady-state models were facilitated by two Python scripts, which are listed in Appendices B and C.

Figure 2 shows ΔT in the tuff versus distance from the drum at various times. It can be seen that the profile has a large gradient building up near the drum interface during the beginning years. However, the profile begins to flatten out as time goes beyond 50 years (after four half lives of tritium decay), when the power has weakened significantly and the heat has conducted throughout the surrounding tuff.

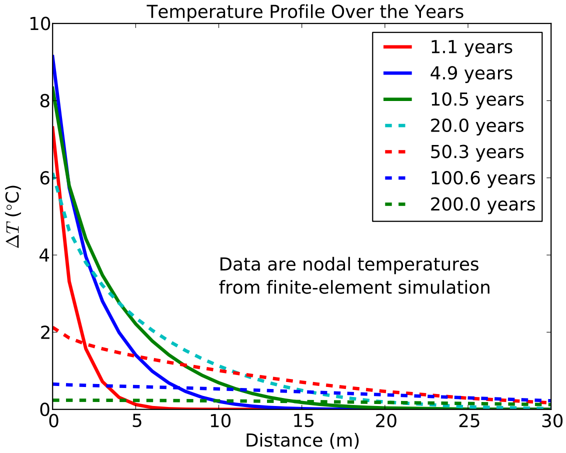

Figure 2: Temperature profile in surrounding tuff.

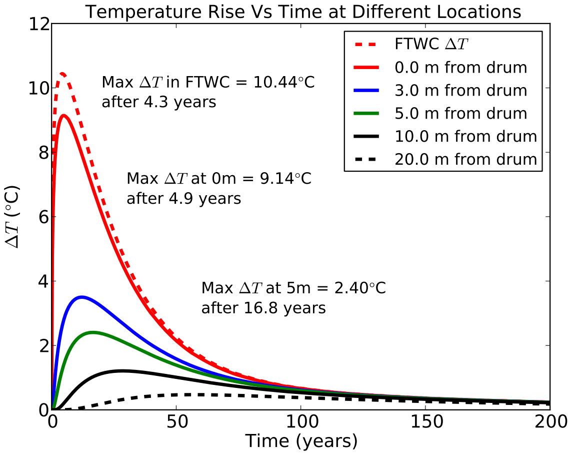

Figure 3: Temperature history at various distances.

Figure 3 shows the calculated ΔT history in the FTWC as well as in the tuff at various distances from the drum. The temperature rise in the FTWC is obtained by adding the ΔT in the tuff at the drum interface and the ΔT across the FTWC/skirt/drum structure calculated from the steady-state model. As can be seen in the plots, the maximum heat up in the FTWC is around 10°C, which occurs at 4.3 years after burial. Peak ΔT at the drum is about 9°C. At 5 m away, the maximum ΔT reaches 2.4°C after roughly 17 years. Yet further away, at 10 m, the maximum ΔT is 1.2°C after 29 years.

3.2 Pressure Build-up

The FTWC is designed to a maximum allowable working pressure (MAWP) of 300 psig [2]. As tritium decays, pressure will build up in the container due to gases (mainly hydrogen) generated by radiolysis in addition to the helium decay product. We follow the method for calculating radiolytic hydrogen generation and the resulting pressure in the FTWC as given in a spreadsheet by Roy Michelotti [9]. The calculation assumes a void volume of 60% and that the initial temperature in the FTWC is that of the tuff at buried depth, which is 9°C, a value estimated from the subsurface temperature data in a 1976 LASL report by Reiter et al [8]. The FTWC temperature increases over its initial value according to the time-dependent ΔT shown in figure 3. A Python program, listed in Appendix D, was written for the pressure calculation with a time-dependent temperature. Figure 4 shows the pressure build-up as a function of time to 200 years, long after all tritium has decayed away. The maximum calculated pressure is 276.5 psig.

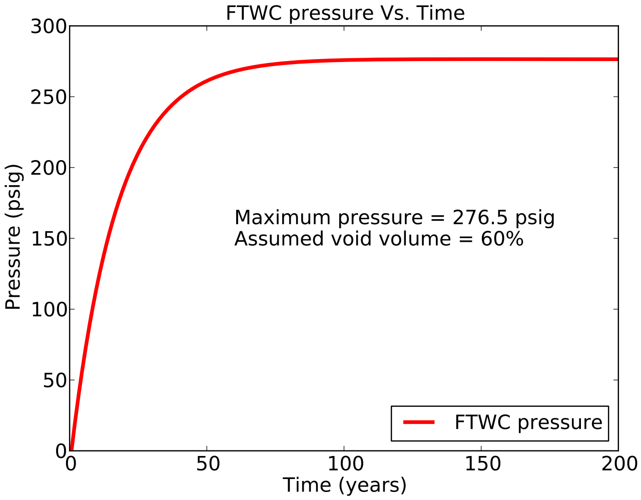

Figure 4: Pressure build-up in the FTWC as a result of radiolysis.

4 Conclusion

This study shows that the maximum temperature rise in the FTWC is about 10°C. This heat up is not expected to adversely affect the mechanical properties of the stainless steel FTWC and its integrity as a waste container. The ΔT in the tuff immediately next to the drum reaches a maximum of 9°C. However, at 5 m away, the peak ΔT drops to 2.4°C. Further away, at 10 m, the maximum ΔT is 1.2°C. The FTWC thermal output therefore should have minimal detrimental effects on the surrounding waste containers.

Pressure build-up in the FTWC as a result of radiolysis in the tritium waste has also been calculated, with the thermal load taken into consideration. The maximum pressure is 276.5 psig after all tritium has decayed. Therefore, as-buried FTWCs will remain below the design pressure limit of 300 psig under all expected conditions.

References

[1] Michelotti, Roy, FTWC Thermal Loading Scope of Work.

[2] Engineering drawing, Flanged Tritium Waste Container, Los Alamos National Laboratory, Dwg. No. 11669-001 rev. 1.

[3] Specification sheet, "Procurement Specification for UN 1A2/DOT 7A Type A, Open Head, Stainless Steel Drums," Los Alamos National Laboratory, P&T-SPEC-37, R3.

[4] Incropera, Frank P., and DeWitt, David P., *Fundamentals of Heat and Mass Transfer*, Wiley, New York, 1996, p. 829, 839.

[5] Information about the Abaqus commercial finite-element analysis package can be found at: `http://www.3ds.com/products-services/simulia/portfolio/abaqus/abaqus-portfolio`

[6] Wohletz, Kenneth, "Fractures in welded tuff", *Geological Society of America: Special Paper 408*, 2006, Chapter 2.3.

[7] Lappin, Allen R., "Thermal Conductivity of Silicic Tuffs: Predictive Formalism and Comparison With Preliminary Experimental Results," Sandia National Laboratories, Project No. SAND80-0769, Albuquerque, NM, July 1980.

[8] Reiter, Marshall, Weidman, Charles, Edwards, C.L., and Hartman, Harold, "Subsurface Temperature Data in Jemez Mountains, New Mexico," TR, Los Alamos Scientific Laboratory, Circular 151, Socorro, NM, 1976.

[9] Michelotti, Roy, Speadsheet calculating radiolytic gas generation and pressure build-up in FTWC.

Appendix A

The Abaqus input file for the transient axisymmetric model is listed below.

```
*Heading
 FTWC transient
** Job name: jt50_decay Model name: Model-1
** Generated by: Abaqus/CAE 6.12-1
*Preprint, echo=NO, model=NO, history=NO, contact=NO
**
** PARTS
**
*Part, name=Part-1
*End Part
**
**
** ASSEMBLY
**
*Assembly, name=Assembly
**
*Instance, name=Part-1-1, part=Part-1
*Node
      1,   50.6599998,        1.097
      2,   49.6599998,        1.097
      3,   48.6599998,        1.097
      4,   47.6599998,        1.097
      5,   46.6599998,        1.097
      6,   45.6599998,        1.097
      7,   44.6599998,        1.097
      8,   43.6599998,        1.097
      9,   42.6599998,        1.097
     10,   41.6599998,        1.097
     11,   40.6599998,        1.097
     12,   39.6599998,        1.097
     13,   38.6599998,        1.097
     14,   37.6599998,        1.097
     15,   36.6599998,        1.097
     16,   35.6599998,        1.097
     17,   34.6599998,        1.097
     18,   33.6599998,        1.097
     19,   32.6599998,        1.097
     20,   31.6599998,        1.097
     21,   30.6599998,        1.097
     22,   29.6599998,        1.097
     23,   28.6599998,        1.097
     24,   27.6599998,        1.097
     25,   26.6599998,        1.097
     26,   25.6599998,        1.097
     27,   24.6599998,        1.097
     28,   23.6599998,        1.097
     29,   22.6599998,        1.097
     30,   21.6599998,        1.097
     31,   20.6599998,        1.097
     32,   19.6599998,        1.097
     33,   18.6599998,        1.097
     34,   17.6599998,        1.097
```

```
35,   16.6599998,        1.097
36,   15.6599998,        1.097
37,   14.6599998,        1.097
38,   13.6599998,        1.097
39,   12.6599998,        1.097
40,   11.6599998,        1.097
41,   10.6599998,        1.097
42,   9.65999985,        1.097
43,   8.65999985,        1.097
44,   7.65999985,        1.097
45,   6.65999985,        1.097
46,   5.65999985,        1.097
47,   4.65999985,        1.097
48,   3.66000009,        1.097
49,   2.66000009,        1.097
50,   1.65999997,        1.097
51,  0.660000026,        1.097
52,   50.6599998,  0.100000001
53,   49.6599998,  0.100000001
54,   48.6599998,  0.100000001
55,   47.6599998,  0.100000001
56,   46.6599998,  0.100000001
57,   45.6599998,  0.100000001
58,   44.6599998,  0.100000001
59,   43.6599998,  0.100000001
60,   42.6599998,  0.100000001
61,   41.6599998,  0.100000001
62,   40.6599998,  0.100000001
63,   39.6599998,  0.100000001
64,   38.6599998,  0.100000001
65,   37.6599998,  0.100000001
66,   36.6599998,  0.100000001
67,   35.6599998,  0.100000001
68,   34.6599998,  0.100000001
69,   33.6599998,  0.100000001
70,   32.6599998,  0.100000001
71,   31.6599998,  0.100000001
72,   30.6599998,  0.100000001
73,   29.6599998,  0.100000001
74,   28.6599998,  0.100000001
75,   27.6599998,  0.100000001
76,   26.6599998,  0.100000001
77,   25.6599998,  0.100000001
78,   24.6599998,  0.100000001
79,   23.6599998,  0.100000001
80,   22.6599998,  0.100000001
81,   21.6599998,  0.100000001
82,   20.6599998,  0.100000001
83,   19.6599998,  0.100000001
84,   18.6599998,  0.100000001
85,   17.6599998,  0.100000001
86,   16.6599998,  0.100000001
87,   15.6599998,  0.100000001
88,   14.6599998,  0.100000001
89,   13.6599998,  0.100000001
90,   12.6599998,  0.100000001
```

```
     91,   11.6599998,  0.100000001
     92,   10.6599998,  0.100000001
     93,   9.65999985,  0.100000001
     94,   8.65999985,  0.100000001
     95,   7.65999985,  0.100000001
     96,   6.65999985,  0.100000001
     97,   5.65999985,  0.100000001
     98,   4.65999985,  0.100000001
     99,   3.66000009,  0.100000001
    100,   2.66000009,  0.100000001
    101,   1.65999997,  0.100000001
    102,  0.660000026,  0.100000001
*Element, type=DCAX4
 1,   1,   2,  53,  52
 2,   2,   3,  54,  53
 3,   3,   4,  55,  54
 4,   4,   5,  56,  55
 5,   5,   6,  57,  56
 6,   6,   7,  58,  57
 7,   7,   8,  59,  58
 8,   8,   9,  60,  59
 9,   9,  10,  61,  60
10,  10,  11,  62,  61
11,  11,  12,  63,  62
12,  12,  13,  64,  63
13,  13,  14,  65,  64
14,  14,  15,  66,  65
15,  15,  16,  67,  66
16,  16,  17,  68,  67
17,  17,  18,  69,  68
18,  18,  19,  70,  69
19,  19,  20,  71,  70
20,  20,  21,  72,  71
21,  21,  22,  73,  72
22,  22,  23,  74,  73
23,  23,  24,  75,  74
24,  24,  25,  76,  75
25,  25,  26,  77,  76
26,  26,  27,  78,  77
27,  27,  28,  79,  78
28,  28,  29,  80,  79
29,  29,  30,  81,  80
30,  30,  31,  82,  81
31,  31,  32,  83,  82
32,  32,  33,  84,  83
33,  33,  34,  85,  84
34,  34,  35,  86,  85
35,  35,  36,  87,  86
36,  36,  37,  88,  87
37,  37,  38,  89,  88
38,  38,  39,  90,  89
39,  39,  40,  91,  90
40,  40,  41,  92,  91
41,  41,  42,  93,  92
42,  42,  43,  94,  93
43,  43,  44,  95,  94
```

```
44,  44,  45,  96,  95
45,  45,  46,  97,  96
46,  46,  47,  98,  97
47,  47,  48,  99,  98
48,  48,  49, 100,  99
49,  49,  50, 101, 100
50,  50,  51, 102, 101
*Nset, nset=Set-1, generate
   1,  102,    1
*Elset, elset=Set-1, generate
  1,  50,   1
** Section: tuff
*Solid Section, elset=Set-1, material=tuff
,
*End Instance
**
*Nset, nset=Set-5, instance=Part-1-1
  1, 52
*Elset, elset=Set-5, instance=Part-1-1
 1,
*Nset, nset=Set-6, instance=Part-1-1, generate
   1,  102,    1
*Elset, elset=Set-6, instance=Part-1-1, generate
  1,  50,   1
*Nset, nset=node01, instance=Part-1-1
 51,
*Nset, nset=node02, instance=Part-1-1
 50,
*Nset, nset=node03, instance=Part-1-1
 49,
*Nset, nset=node04, instance=Part-1-1
 48,
*Nset, nset=node05, instance=Part-1-1
 47,
*Nset, nset=node06, instance=Part-1-1
 46,
*Nset, nset=node07, instance=Part-1-1
 45,
*Nset, nset=node08, instance=Part-1-1
 44,
*Nset, nset=node09, instance=Part-1-1
 43,
*Nset, nset=node10, instance=Part-1-1
 42,
*Nset, nset=node11, instance=Part-1-1
 41,
*Nset, nset=node12, instance=Part-1-1
 40,
*Nset, nset=node13, instance=Part-1-1
 39,
*Nset, nset=node14, instance=Part-1-1
 38,
*Nset, nset=node15, instance=Part-1-1
 37,
*Nset, nset=node16, instance=Part-1-1
 36,
*Nset, nset=node17, instance=Part-1-1
 35,
*Nset, nset=node18, instance=Part-1-1
 34,
*Nset, nset=node19, instance=Part-1-1
 33,
*Nset, nset=node20, instance=Part-1-1
 32,
*Nset, nset=node21, instance=Part-1-1
 31,
*Nset, nset=node26, instance=Part-1-1
 26,
*Nset, nset=node31, instance=Part-1-1
 21,
*Nset, nset=node36, instance=Part-1-1
 16,
*Nset, nset=node41, instance=Part-1-1
 11,
*Nset, nset=node46, instance=Part-1-1
 6,
*Nset, nset=node51, instance=Part-1-1
 1,
*Elset, elset=_Surf-2_S2, internal, instance=Part-1-1
 50,
*Surface, type=ELEMENT, name=Surf-2
_Surf-2_S2, S2
*End Assembly
*Amplitude, name="q’’decay2", definition=DECAY
             0.,              1.,              0.,      560520956.
**
** MATERIALS
**
*Material, name=tuff
*Conductivity
 0.2,
*Density
1800.,
*Specific Heat
1000.,
**
** PREDEFINED FIELDS
**
** Name: Predefined Field-1   Type: Temperature
*Initial Conditions, type=TEMPERATURE
Set-6, 0.
** ----------------------------------------------------------------
**
** STEP: transient
**
*Step, name=transient, inc=1000
FTWC transient
*Heat Transfer, end=PERIOD, deltmx=0.1
1., 6.3072e+09, 0.5, 6.3072e+09,
**
** BOUNDARY CONDITIONS
**
```

```
** Name: T_inf Type: Temperature
*Boundary
Set-5, 11, 11
**
** LOADS
**
** Name: flux   Type: Surface heat flux
*Dsflux, amplitude="q’’decay2"
Surf-2, S, 1.566
**
** OUTPUT REQUESTS
**
*Restart, write, frequency=0
**
** FIELD OUTPUT: F-Output-1
**
*Output, field, variable=PRESELECT
*Output, history, frequency=0
*End Step
```

Appendix B

This is an Abaqus Python script to extract nodal temperatures from the output database and write their values as a function of time to a text file for post-processing calculations and plotting.

```
from odbAccess import *
import glob
import pdb

def time_temp(odb_name):
    """Pulls temperature data from the thermal, axisymmetric FTWC simulation at every
       time step

    """

    odb = openOdb(odb_name)

    #string name of text file to write results to
    s_fname = odb_name[:-4] + ’_data.txt’


    s_file = open(s_fname,’w’)

    #grabs the last step of the analysis, which is going to be the load step
    #we’re interested in

    step = odb.steps[odb.steps.keys()[-1]]

    #loop over the frames in the step and pull the temp data from an element.

    s_file.write(’# Time          temp (degrees C) \n’)

    #pdb.set_trace()
    roi01 = odb.rootAssembly.nodeSets[’NODE01’]
    roi02 = odb.rootAssembly.nodeSets[’NODE02’]
    roi03 = odb.rootAssembly.nodeSets[’NODE03’]
    roi04 = odb.rootAssembly.nodeSets[’NODE04’]
    roi05 = odb.rootAssembly.nodeSets[’NODE05’]
    roi06 = odb.rootAssembly.nodeSets[’NODE06’]
    roi07 = odb.rootAssembly.nodeSets[’NODE07’]
    roi08 = odb.rootAssembly.nodeSets[’NODE08’]
    roi09 = odb.rootAssembly.nodeSets[’NODE09’]
    roi10 = odb.rootAssembly.nodeSets[’NODE10’]
    roi11 = odb.rootAssembly.nodeSets[’NODE11’]
    roi12 = odb.rootAssembly.nodeSets[’NODE12’]
    roi13 = odb.rootAssembly.nodeSets[’NODE13’]
    roi14 = odb.rootAssembly.nodeSets[’NODE14’]
    roi15 = odb.rootAssembly.nodeSets[’NODE15’]
    roi16 = odb.rootAssembly.nodeSets[’NODE16’]
    roi17 = odb.rootAssembly.nodeSets[’NODE17’]
    roi18 = odb.rootAssembly.nodeSets[’NODE18’]
    roi19 = odb.rootAssembly.nodeSets[’NODE19’]
    roi20 = odb.rootAssembly.nodeSets[’NODE20’]
    roi21 = odb.rootAssembly.nodeSets[’NODE21’]
    roi26 = odb.rootAssembly.nodeSets[’NODE26’]
    roi31 = odb.rootAssembly.nodeSets[’NODE31’]
    roi36 = odb.rootAssembly.nodeSets[’NODE36’]
    roi41 = odb.rootAssembly.nodeSets[’NODE41’]
    roi46 = odb.rootAssembly.nodeSets[’NODE46’]
    roi51 = odb.rootAssembly.nodeSets[’NODE51’]

    #I want to loop over all frames and get max(nt11) for each frame
    #Loop over all of the frames in the output database

    for frame in step.frames:

        #grab the time of the current frame
        time = str(frame.frameValue)

        temp01 = frame.fieldOutputs[’NT11’].getSubset(region=roi01).values[0].data
        temp02 = frame.fieldOutputs[’NT11’].getSubset(region=roi02).values[0].data
        temp03 = frame.fieldOutputs[’NT11’].getSubset(region=roi03).values[0].data
        temp04 = frame.fieldOutputs[’NT11’].getSubset(region=roi04).values[0].data
        temp05 = frame.fieldOutputs[’NT11’].getSubset(region=roi05).values[0].data
        temp06 = frame.fieldOutputs[’NT11’].getSubset(region=roi06).values[0].data
        temp07 = frame.fieldOutputs[’NT11’].getSubset(region=roi07).values[0].data
        temp08 = frame.fieldOutputs[’NT11’].getSubset(region=roi08).values[0].data
        temp09 = frame.fieldOutputs[’NT11’].getSubset(region=roi09).values[0].data
        temp10 = frame.fieldOutputs[’NT11’].getSubset(region=roi10).values[0].data
        temp11 = frame.fieldOutputs[’NT11’].getSubset(region=roi11).values[0].data
        temp12 = frame.fieldOutputs[’NT11’].getSubset(region=roi12).values[0].data
        temp13 = frame.fieldOutputs[’NT11’].getSubset(region=roi13).values[0].data
        temp14 = frame.fieldOutputs[’NT11’].getSubset(region=roi14).values[0].data
        temp15 = frame.fieldOutputs[’NT11’].getSubset(region=roi15).values[0].data
        temp16 = frame.fieldOutputs[’NT11’].getSubset(region=roi16).values[0].data
        temp17 = frame.fieldOutputs[’NT11’].getSubset(region=roi17).values[0].data
        temp18 = frame.fieldOutputs[’NT11’].getSubset(region=roi18).values[0].data
        temp19 = frame.fieldOutputs[’NT11’].getSubset(region=roi19).values[0].data
        temp20 = frame.fieldOutputs[’NT11’].getSubset(region=roi20).values[0].data
        temp21 = frame.fieldOutputs[’NT11’].getSubset(region=roi21).values[0].data
        temp26 = frame.fieldOutputs[’NT11’].getSubset(region=roi26).values[0].data
        temp31 = frame.fieldOutputs[’NT11’].getSubset(region=roi31).values[0].data
        temp36 = frame.fieldOutputs[’NT11’].getSubset(region=roi36).values[0].data
        temp41 = frame.fieldOutputs[’NT11’].getSubset(region=roi41).values[0].data
        temp46 = frame.fieldOutputs[’NT11’].getSubset(region=roi46).values[0].data
        temp51 = frame.fieldOutputs[’NT11’].getSubset(region=roi51).values[0].data

        s_file.write(time + ’, ’ + str(temp01) + ’,’ + str(temp02) + ’,’ +
                str(temp03) + ’,’ + str(temp04) + ’,’ + str(temp05) + ’,’ +
                str(temp06) + ’,’ + str(temp07) + ’,’ + str(temp08) + ’,’ +
                str(temp09) + ’,’ + str(temp10) + ’,’ + str(temp11) + ’,’ +
                str(temp12) + ’,’ + str(temp13) + ’,’ + str(temp14) + ’,’ +
                str(temp15) + ’,’ + str(temp16) + ’,’ + str(temp17) + ’,’ +
                str(temp18) + ’,’ + str(temp19) + ’,’ + str(temp20) + ’,’ +
                str(temp21) + ’,’ + str(temp26) + ’,’ + str(temp31) + ’,’ +
                str(temp36) + ’,’ + str(temp41) + ’,’ + str(temp46) + ’,’ +
                str(temp51) + ’\n’)

    #close out the file
    s_file.close()

time_temp(’jt50_decay.odb’)
```

Appendix C

This Python script calculates the FTWC temperature rise by adding the steady-state temperature rise to the transient finite-element results. Two figures are generated, one for ΔT vs. time at different locations and another for ΔT profiles at different times.

```
import matplotlib.pyplot as plt
import numpy as np

# Import the abaqus data as an array
data = np.loadtxt(open("jt50_decay_data.txt"),delimiter=",")

# Breaking the data into time and temperature components
time = data[:,0]                  # seconds
n0t = data[:,1]                   # degrees C
n1t = data[:,2]                   # degrees C
n3t = data[:,4]                   # degrees C
n5t = data[:,6]                   # degrees C
n10t = data[:,11]                 # degrees C
n20t = data[:,21]                 # degrees C

# nodal distances
n0d = 0.0                         # meters
n1d = 1.0                         # meters
n3d = 3.0                         # meters
n5d = 5.0                         # meters
n10d = 10.0                       # meters
n20d = 20.0                       # meters

# Finding the max temps at 0m and 5m
# Also finding their indexes so I can determine the time
m0nt = np.amax(n0t)               # max nodal temp (node 0)
i0mnt = np.argmax(n0t)            # index of max nodal temp (node 0)
m5nt = np.amax(n5t)               # max nodal temp (node 5)
i5mnt = np.argmax(n5t)            # index of max nodal temp (node 5)
To = 1.68                         # initial temp change across FTWC system

# converting from seconds to years
years = np.divide(time,31536000)

# time indexes
t1 = 145        # ˜1 yr
t2 = 191        # ˜5 yr
t3 = 208        # ˜10 yr
t4 = 232        # ˜20 yr
t5 = 296        # ˜50 yr
t6 = 324        # ˜100 yr
t7 = 333        # ˜200 yr

# using the indexes from above to find the time (in years)
py1 = years[t1]
py2 = years[t2]
py3 = years[t3]
py4 = years[t4]
py5 = years[t5]
py6 = years[t6]
```

```
py7 = years[t7]

# Profile data, using the same indexes listed above
prof1 = data[t1,1:28]
prof2 = data[t2,1:28]
prof3 = data[t3,1:28]
prof4 = data[t4,1:28]
prof5 = data[t5,1:28]
prof6 = data[t6,1:28]
prof7 = data[t7,1:28]
# print prof1

#
dist = [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,25,30,35,40,45,50]

# tritium half life - 12.32 years
thl = 388523520
texp = np.divide(time,thl)
thlm = (1./2.)**texp
Tdecay = np.multiply(To,thlm)      # steady state temperature decay
TFTWC = np.add(n0t,Tdecay)         # temp inside FTWC
mTF = np.amax(TFTWC)               # max temp inside FTWC
imTF = np.argmax(TFTWC)            # index of max FTWC temp
t0mnt = years[i0mnt]
t5mnt = years[i5mnt]
tmTF = years[imTF]

plt.figure(1)
# Plotting the temperature profiles at various years
plt.plot(dist,prof1,'r',linewidth=3.0,label='%.1f years' % py1)
plt.plot(dist,prof2,'b',linewidth=3.0,label='%.1f years' % py2)
plt.plot(dist,prof3,'g',linewidth=3.0,label='%.1f years' % py3)
plt.plot(dist,prof4,'c--',linewidth=3.0,label='%.1f years' % py4)
plt.plot(dist,prof5,'r--',linewidth=3.0,label='%.1f years' % py5)
plt.plot(dist,prof6,'b--',linewidth=3.0,label='%.1f years' % py6)
plt.plot(dist,prof7,'g--',linewidth=3.0,label='%.1f years' % py7)

# plotting parameters
plt.xlabel('Distance (m)',fontsize=16)
plt.ylabel(r'$\Delta T$ ($\degree$C)',fontsize=16)
plt.title('Temperature Profile Over the Years',fontsize=16)
plt.text(10,3.6,'Data are nodal temperatures',fontsize=16)
plt.text(10,3.0,'from finite-element simulation',fontsize=16)
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
# plt.ylim([-0.05,10])
plt.xlim([0,30])
plt.rc('legend',fontsize=16)
plt.legend()
plt.savefig('50m_profile.pdf')

plt.figure(2)
plt.plot(years,TFTWC,'r--',linewidth=3.0,label='FTWC $\Delta T$')
plt.plot(years,n0t,'r',linewidth=3.0,label='%.1f m from drum' % n0d)
plt.plot(years,n3t,'b',linewidth=3.0,label='%.1f m from drum' % n3d)
plt.plot(years,n5t,'g',linewidth=3.0,label='%.1f m from drum' % n5d)
plt.plot(years,n10t,'k',linewidth=3.0,label='%.1f m from drum' % n10d)
plt.plot(years,n20t,'k--',linewidth=3.0,label='%.1f m from drum' % n20d)

# Plotting paramaters
plt.xlabel('Time (years)',fontsize=16)
plt.ylabel(r'$\Delta T$ ($\degree$C)',fontsize=16)
plt.title('Temperature Rise Vs Time at Different Locations',fontsize=16)
plt.text(60,3.6,r'Max $\Delta T$ at 5m = %.2f$\degree$C' % m5nt,fontsize=14)
plt.text(60,3.0,'after %.1f years' % t5mnt,fontsize=14)
plt.text(30,7.0,r'Max $\Delta T$ at 0m = %.2f$\degree$C' % m0nt,fontsize=14)
plt.text(30,6.4,'after %.1f years' % t0mnt,fontsize=14)
plt.text(20,10.0,r'Max $\Delta T$ in FTWC = %.2f$\degree$C' % mTF,fontsize=14)
plt.text(20,9.4,'after %.1f years' % tmTF,fontsize=14)
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
plt.xlim([0,200])
plt.rc('legend',fontsize=14)
plt.legend()
plt.savefig('50m_FTWC_temp_time_dist.pdf')

#plt.show()
```

Appendix D

This Python script calculates and plots the pressure build-up as a function of time.

```
import matplotlib.pyplot as plt
import numpy as np

# Import the abaqus data as an array
data = np.loadtxt(open("jt50_decay_data.txt"),delimiter=",")


# Breaking the data into time and temperature components
t = data[:,0]                         # seconds (decaying heat flux)
n0t = data[:,1]                       # node 1 deltaT (decaying heat flux)
To = 1.68                             # initial deltaT for FTWC
Ta = 282                              # initial ambient (tuff) temperature
dr = np.shape(data)[0]                # number of rows in data

# converting from seconds to years
years = np.divide(t,31536000)

thalf = 3600*24*365*12.32             # Tritium half life in seconds
td = thalf/(np.log(2.))               # decay time in seconds
texp = np.divide(t,thalf)             # decay exponent
thlm = (1./2.)**texp                  # half life multiplier
Tdecay = np.multiply(To,thlm)         # steady state temperature decay
Tf = n0t + Tdecay + Ta                # deltaT inside FTWC

m0 = 10                               # initial mass of tritium
mw = 3.015                            # molecular weight (g/mole)
ebeta = 5700                          # energy per decay (eV)
nbH2 = 0.005                          # H2 molecules per decay (molecules/eV)
R = 1.2062                            # gas constant (psi-L/mole-K)
vf = 193.29                           # volume of FTWC (L)
vp = 60.                              # void percentage
vol = vf*vp*.01                       # open volume of FTWC (L)
Pw = 0.463                            # water vapor pressure (psi)
Patm = 11.218                         # atmospheric pressure (psi)

# initializing for loop
P = np.zeros(dr)

for i in range(dr):
    nhe = m0*(1-np.exp(-t[i]/td))/mw        # number of moles of He
    nh2 = nhe*ebeta*nbH2
    ntot = nhe+nh2
    P[i] = ntot*R*Tf[i]/vol + Pw - Patm

mP = np.amax(P)

# Plotting the pressure versus time in years
plt.plot(years,P,’r’,linewidth=3.0,label=’FTWC pressure’)

# plotting parameters
plt.xlabel(’Time (years)’,fontsize=16)
plt.ylabel(’Pressure (psig)’,fontsize=16)
plt.title('FTWC pressure Vs. Time',fontsize=16)
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
plt.xlim([0,200])
plt.ylim([0,300])
plt.rc('legend',fontsize=16)
plt.legend(loc=4)
plt.text(60,160,'Maximum pressure = %.1f psig' % mP,fontsize=16)
plt.text(60,145,'Assumed void volume = %.0f%%' % vp,fontsize=16)

plt.savefig('pressure9C.pdf')
#plt.show()
```